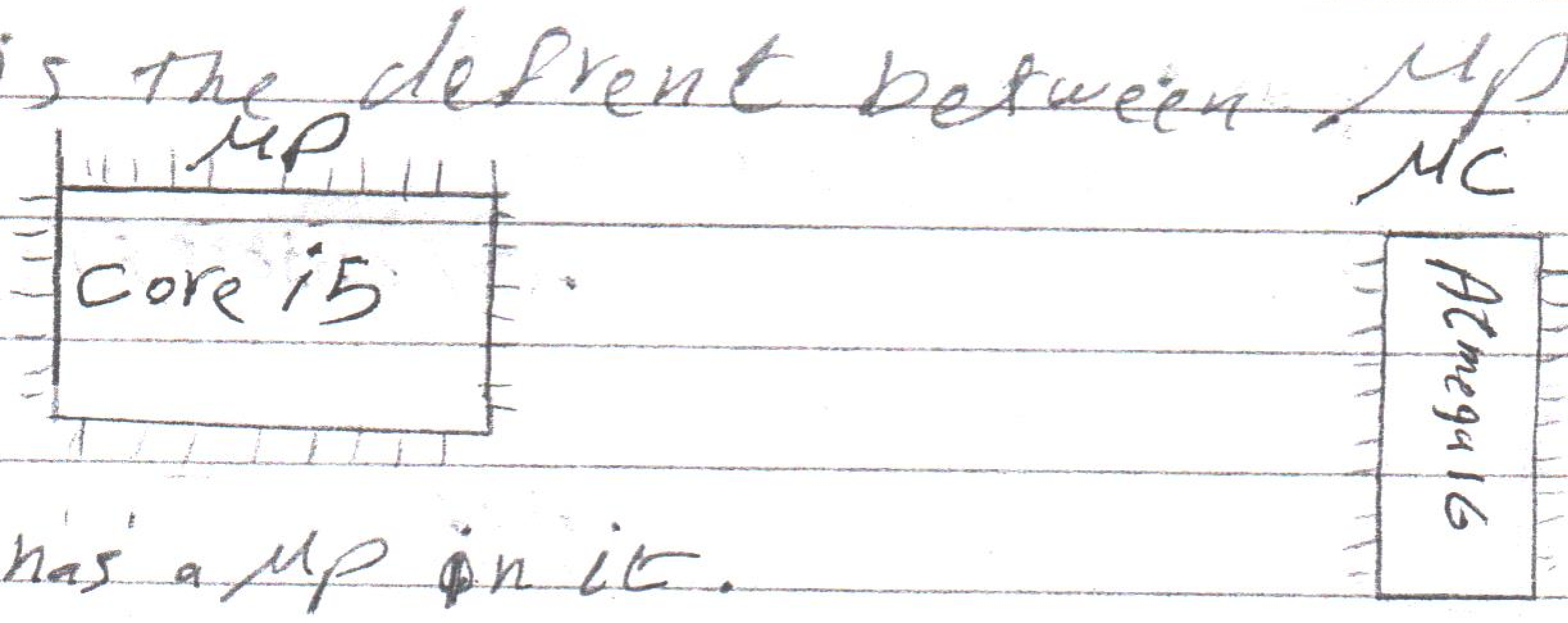
PAGE 30/1/2016
DATE

# Introdultion

what is Embedded system ?

it is soft ware on hardware to do cearrain Perpose :

what is the defrent between MP and MC ?



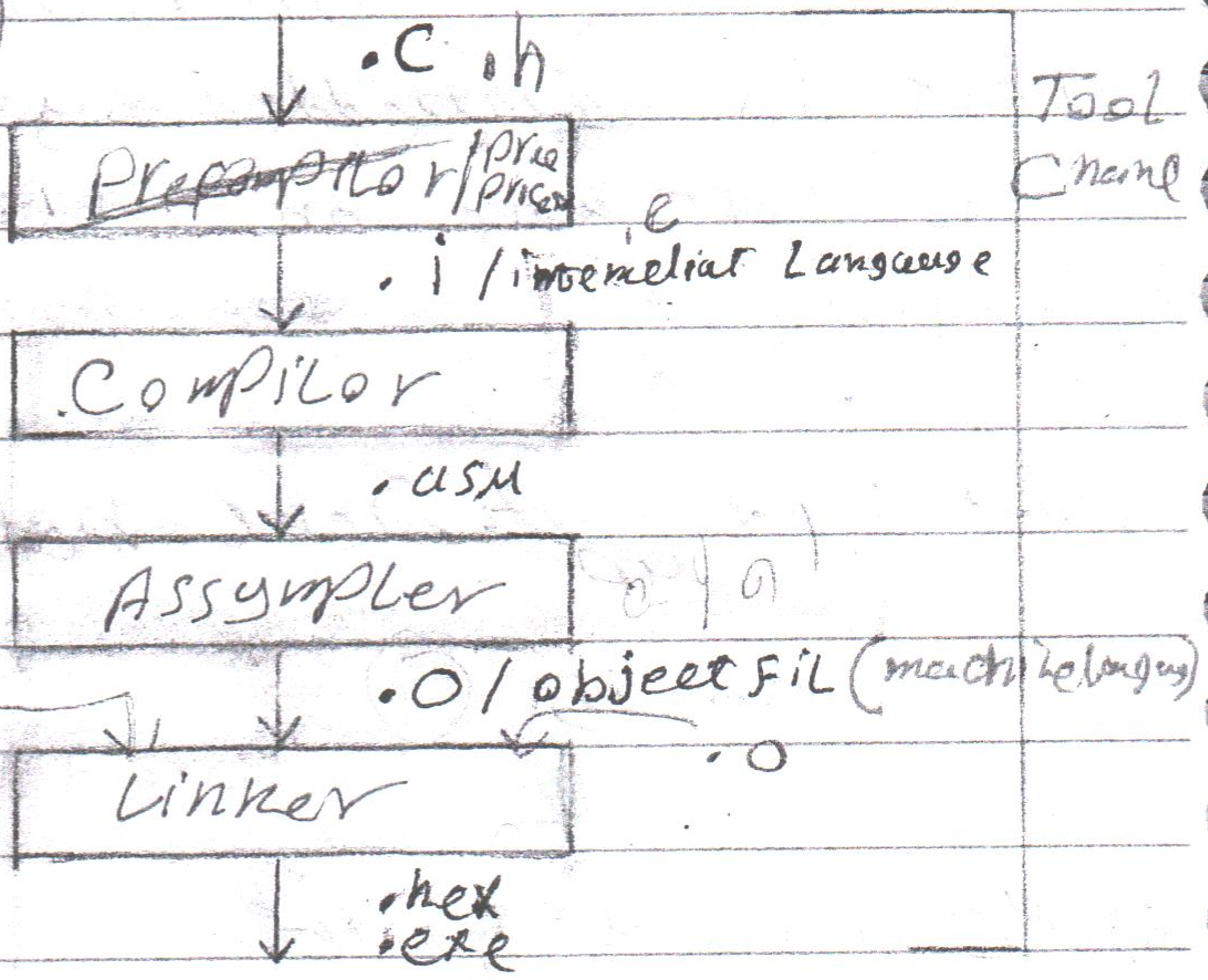
① MC has a MP in it.
② The MP in MC is very small and sheep.
③ MP Like core i5 cost around 1000 L.E.
④ MC do one or some fixed functions.

## S.W Constrans

① Cost
② optimized Code(memory) كل سطر بكلفة ...... 0
③ Timing-(Soft real Time system, Hard realtime system)
④ Realiability. (Can be TrusT~~ed~~)
⑤ Power consumption.

⑤ MP General Perpose , MC specific Perpose
⑥ MP ~~dosen't~~ hase memory or Prepherals, MC hase
⑦
⑧
⑨

## • Bild Process.



* كل Target ليها الـ Tool Chane مختلفة عن التانية

* كلهم بيكتبو C (Embedd C)

وكلهم في الآخر بيحولوا لـ machine Language

* Target: The H.w That what I'm develop for (Atmil)
* HOST: The machine I'm develop on it (labtop).
* Compilor: The Target and The host is The sare.
* CROSS compilor: Target diffrent from The host.

memory

| Volatile (RAM) | Floating gate Non Volatile (RoM) |
|---|---|

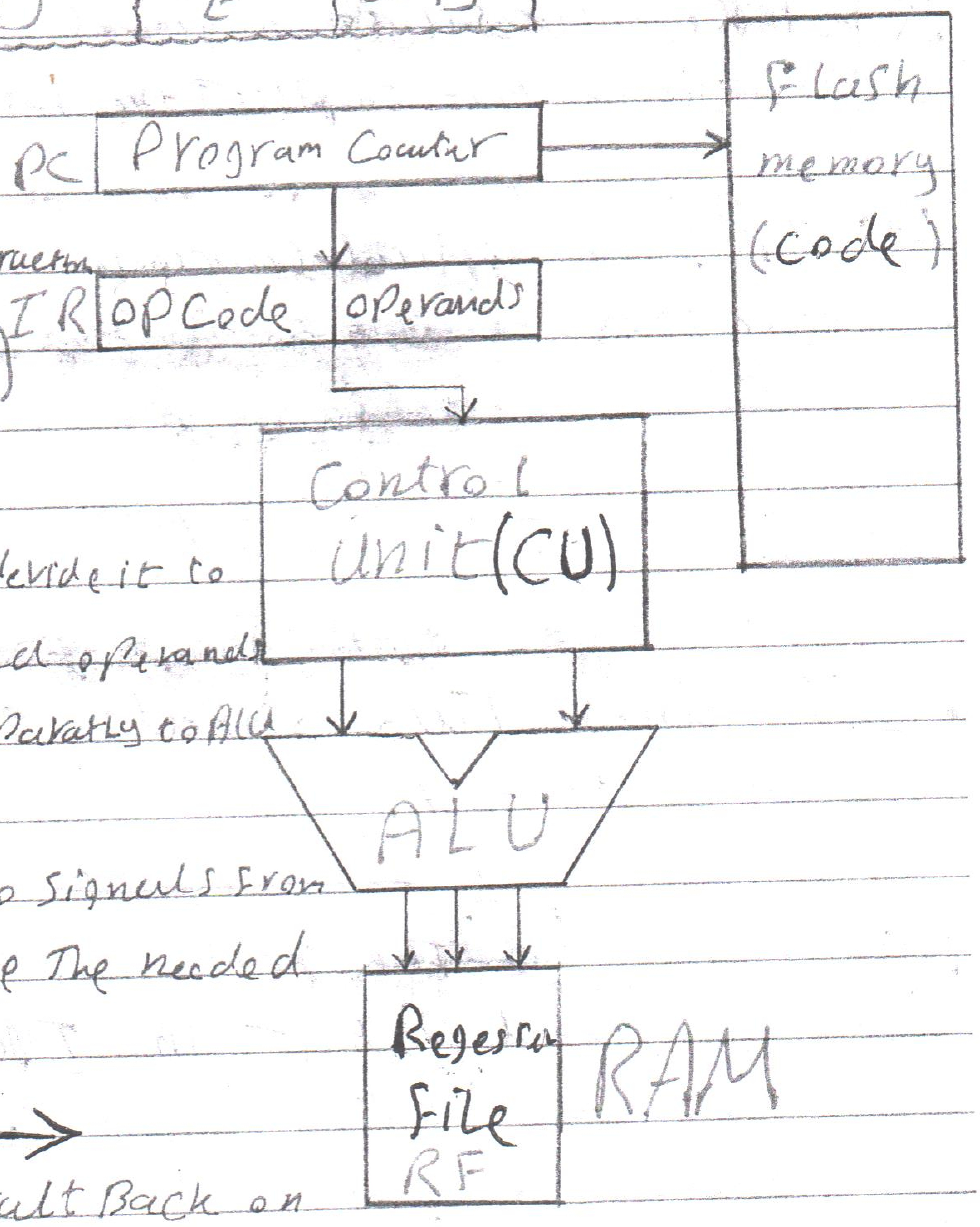
- DRAM: Need refresh cycle
- SRAM: Dosn't need refreshc.
- Flash: Code memory
- EEPROM:

| .DRAM | SRAM | Flash | E²PRoM |
|---|---|---|---|
|  |  | NAND | NoR |
| Power ↑ | Power ↓ | | |
| COST ↓ | COST ↑ | COST ▮ ↑ | COST ▮ ↓ |
| speed ↓ | speed ↑ | speed ▮ ↑ | speed ▮ ↓ |
| capasitors | Transistors | لازم تمسح كلها | ممكن تمسح بايت Byte |

عند ما افصل الكهرباء بتضيع الـ Data

# Excution Life cycle:

| | | | |
|---|---|---|---|
| F | D | E | WB |



① FETCh →

PC point To The instruction set and put it in (instruction Regester)

② DeCode ⇄

instruction regester devide it to operation code and operands and send Them eeparatly to ALU

③ excute →

ALU Take The Two Signals from The CU and excute The needed calculation

④ write Back →

ALU write The result Back on The RF

5/2/2016

- Regesters:

regesters are a memory locations.

RF: Regester file is a general Perpose regester near to ALU take fast

PC, IR: are aspical fancuion regesters.

| RAM |
|---|
| GPR: general Perpose regester |
| SFR: spicial function regester |
| SRAM: static Ram |

- CU Sends Takes Two signals from ALU. That get Them from IR was from PC

PC Point to the next instrution to be excuted and put it in IR That devied Them To Two halfs The first is opcode and the second is operands. Then IR send Them to CU sends two signals to ALU take The opcode and save it, operand writes Them on RAM(RF) and Then make The Calculations and write Them back in RF

# C-Language

high Level — Java, C++, C#

— C_Language.

— Assymply L.

Low level — machine L.

C is an intemediate language

## ① Data Types

| primptive | Drived | user defined |
|---|---|---|
| int | ② Array | struct |
| Char | ① Function | union |
| float | ③ Pointer | enum |
| double | | |
| Void | | |

int size → 4 Byte

Char size → 1 Byte

float size → 4 Byte

double size → 8 Byte

machin dependent
native compiler

GCC
Linux

any Code must hase

| | |
|---|---|
| ① preproccissor Directives | # includ <stdio.h> |
| ② Global Dediration variable | int x ; |
| ③ Main function | int main( )<br>{<br>} |

## ③ operations

| Arithmatic | Bitwise | Relatineal |
|---|---|---|
| + | & | < |
| − | \| | <= |
| * | ^ | > |
| / | ~ | => |
| % | << | == |
| -- | >> | != |
| ++ | | |

## ④ logecal operators

① &&     ② ||     ③ !

## ⑤ Assigment operators

① +=     ② -=     ③ *=

④ /=     ⑤ %=

[illegible] (ex )) Assign

X = X + 1
X += 1 same

## 3 Code ex:3

write c program to conpare if(X != 10).

① Unarry operator : ++, --

② Binary operator : +, -, <

③ Trinary operator: > ? True : Fals

## Post inc, dec

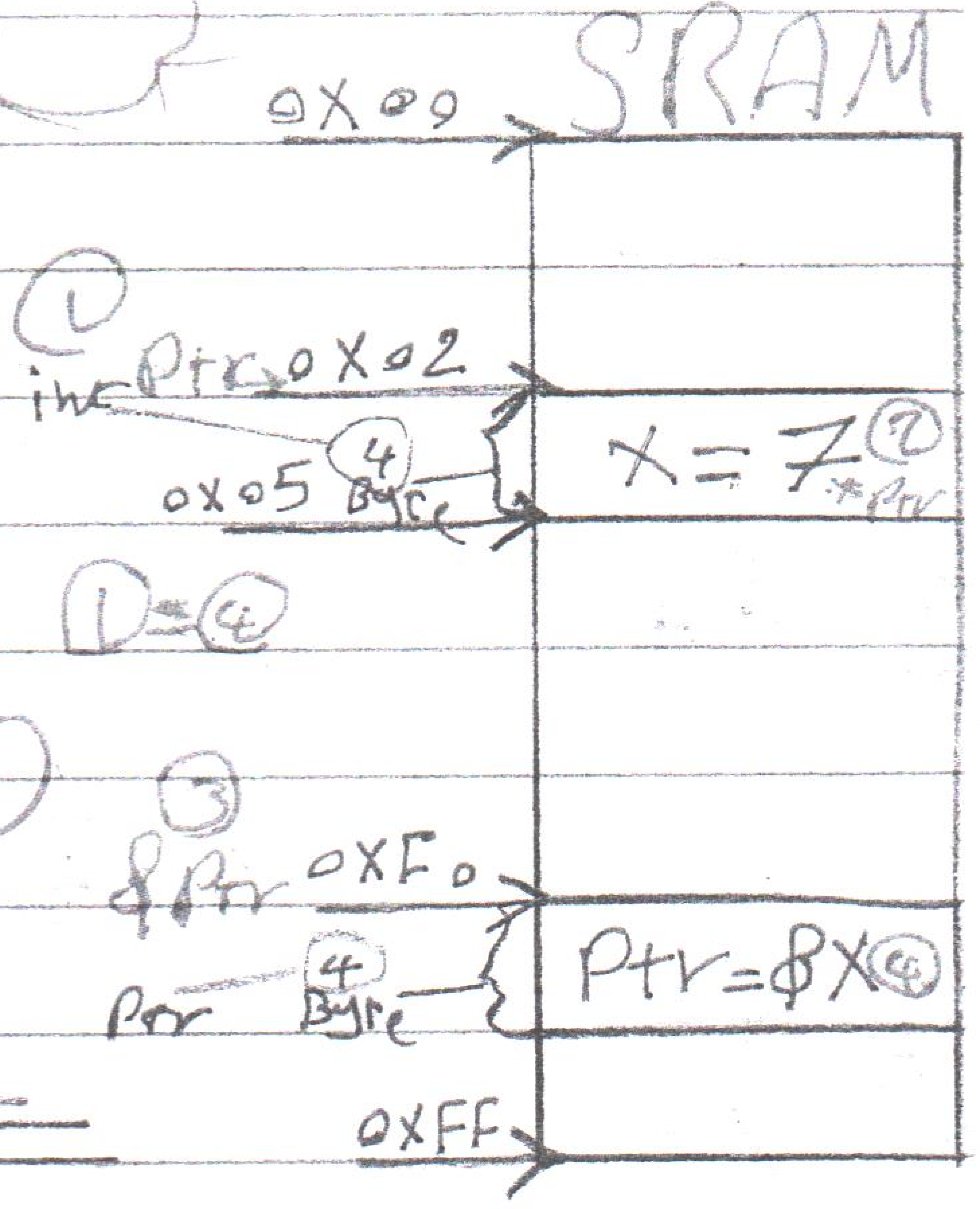
```
int x = 10, y ;
y = x++ ;
printf("%d", x); → 11
Pritf ("%d", y); → 10
```

## pree dec, inc

```
int x = 10, y ;
y = ++x ;
printf("%d", x) ; → 11
printf ("%d", y); → 11
```

## [1] Pointers

```
int x ;
int *ptr = &X ;
*ptr = 7 ;
printf("%d", x) ; → 7
```



ViNB

size بتاع الـ ptr ثابت ٤ بايت

```
② printf("%p", *ptr) ; → 7 (x)
①·① printf ("%p", ptr) ; → 0x02 (&X)
③ printf ("%p", &ptr) ; → 0xF0
② printf ("%p", x ) ; → 7
```

p → hex.

N.B

* The only defrrence between ptr to int and ptr to char is memory STEP

مع العناوين نمر
الناس ساكنه في نفس الشارع، عنوانهم في البطاقة واحد بس مساحات الشقة مختلفة

## Constant Hacking:

N.B
Const int ~~x~~; error

```
Constant int X=10;
X=5; error
```

```
int *Ptr=&X;
    *Ptr=3;
printf("%d",X); → ✓
```

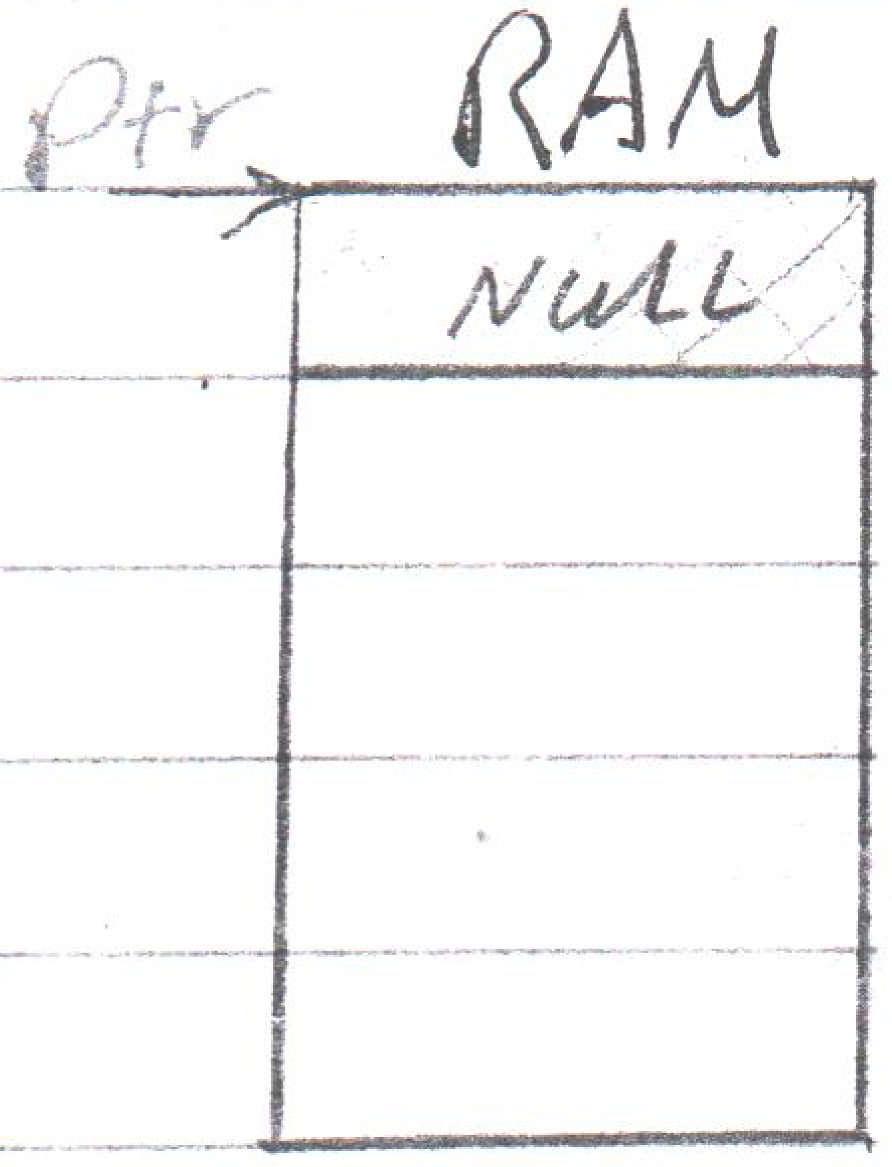
N.B all data Tyeps are singed by default.

unsigned $0 \rightarrow 2^{n}-1$

Signed $-2^{n-1} \rightarrow 2^{n-1}-1$

* كل الميزة ف ال Ptr انى ممكن اغير بيه القيمة اللى بيشاور عليها ⊛ ← البتاعة دى

## wild Pointer



```
int speed;
int *Ptr;        ← مصيبة
int *Ptr=Null;   ← الحل
```

N.B

You Can Create many Ptr ~~for~~ The same memory Location.

☑ Ced: ex4 writ c program Can Add(or) sum Two Variables. (scanf)

PAGE
DATE 6/2/2016

# Switch

```
switch (Var)
      {
         Case 1: ;
            Break;
         Case 2: ;
            Break;
         defult :
            Break;
      }
```

## 4 Notes about switch

① Case grouping :

```
         Case 1:
         Case 2:
         Case 3:
                  printf("  ");
                  Break;
```

② Switch( Flaut, ~~duble~~ )

switah only int and cher.

③ You can't switch conditions.

PAGE

DATE

## ④ Unreachbul Code

```
        switch(X)
        {
unreachable   printf("  ");
              scanf("  ");

          case 1:
               Break;
        }
```

## Map table

```
switch(Y)
  {
      case 1:
      Break;
      default;
  }   Break;
```

Flashmemory

| Y | Addresses |
|---|---|
| 1 | 0xFF0F |
| 2 | 0xFA0C |
| 3 | 0xA02A |
| ~~d~~efult | 0xFF0A |
| | |

(N.B) Some case converted to IF, Else IF
على حسب اللى هيوفر الـ execution time

## ① For loop

For( initLization ; Condition ; itration action )

① initization : ← Code لازم يشوفه الـ

② Condition: الـ For هتعمل Check عليه كل مرة (كل itration) هيشاورلي الأفضل

③ itration action: بيتنفذ في آخر كل itration

(N.B) for(i=0; x>i ; i++) ; → هتخلص كل اللوبس هنا
even if( ); while( ); كأنها شايفة الـ } اللي هي الأخر

## N.B

you can make multi Like:

```
    ( j=1,  ;  &&  ;  J++ )
For ( A=2   ;  ||  ;  i-- )
    ( x=4               )
```

## ② While

```
x = 0;
while(i<10)
{

   i++;
}
```

became For.Loop

## Break;

① works with Looping (for, while)

② works with switch.

## Continue

.makes Loop itration Skiped.

.once forLoop (for Ex) see Continue it will start over again The next itration.

| For | While |
|---|---|
| * بستخدم for لما أكون عارف عدد الـloops اللي أنا عاوزها | * بستخدم While لما يكون في factor خارجي ومش عارف هلف Loop كام مرة |

## infinit LooPS

```
while(1)                    for(;;)
{                            {
}                            }
```

## ③ do...while

```
do
{

}while(Condition);
```

## For Best practice

```
For(i=١٠; i>٠; i--)
```

* في ال ASSymply بتوفر ~~سطر~~ instruction في كل itration.

جميلة

```
int i = ٠ , J=١٠;   Fals   True
    iF ( i++ && J++ )
        {          S.C            printf("%d", i); → ١
        }                         printf("%d", J); → ١٠
```

```
int i=1 , J=١٠
    if ( i++ || J++ )
    {                             printf("%d", i); → 2
    }                             printf("%d", J); → ١٠
```

PAGE 12/12/2016
DATE

address[index]  **[2] Arrays**  Var [ ];

```
int x[10]; → Ruppesh.
int x[10] = {0}; → Zeros كلو
int x[10] = {1,2,3,4,...}; هيحط الاماكن ويعطي
كل واحدة القيم بالترتيب واللي مش initial هيعطيها zero.
```

---

**[5] Code Ex5:**

write C program That init an array and print it with for loop.

---

## Some notes of arrays

1. Name of array if a Constant pointer to The first elemet of array.
2. Size of array [index] mast be defined.
3. arr[0] = x; ✓✓ تنفع
4. int arr[10]. → arr[0] — arr[9]
   arr[10] = 100; over accessing of an array
5. Arr[x]; ✗
6. int x; int y; ...; ✗ arr[10] = {x, y, ...}

Constant Pointer  (N.B Like array)

```
   int x;
int *const ptr = &x;
   ptr++       error
   *ptr = 5;   ✓
```

constant Pointer to an integer

Pointer to constant

```
Const int x;
const int *ptr;
   ptr++       ✓
   *ptr = 20   error
```

Pointer to constant inrger.

## Address [Inded]

index i

size

memory size

int arr[100];

arr [0] → arr + (0 * 4) = arr firstelemt

arr [1] → arr + 1 * 4 = arr+4

arr [2] → arr + 2 * 4 = arr + 8

arr [100] → arr + 4 * 4 = arr+400

## 6 Code Ex6:

write c Programe to scan elements of an array and scan X and compare return the number of element. if not find print "NF" on The Screen.

Flag F=1;

## MultiD"Arrays

### 1 2D array

int Arr[2][3]

2 → # Rows

3 → # Coloms

|   | 0 | 1 | 2 |
|---|---|---|---|
| 0 | arr[0][0] | arr[0][1] | arr[0][2] |
| 1 | arr[1][0] | arr[1][1] | arr[1][2] |

| |
|---|
| |
| arr[0][0] |
| arr[0][1] |
| arr[0][2] |
| arr[1][0] |
| arr[1][1] |
| arr[1][2] |
| |

```
For(i=0;i<2;i++)
  {   For(C=0;C<3;C++)
           {     printf("%d", arr[i][C])
           }
  }
```

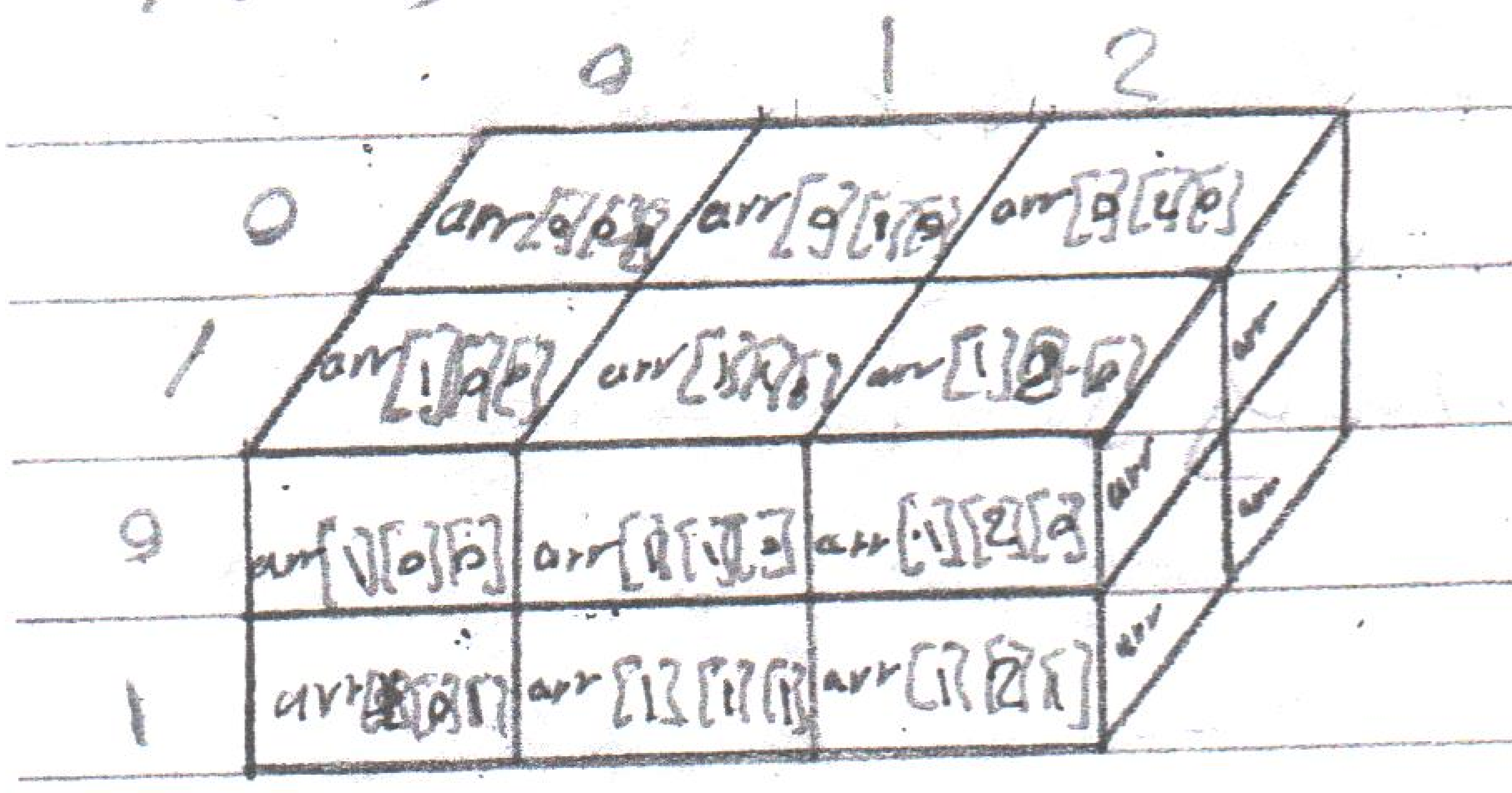
[7] Code [x 7]:

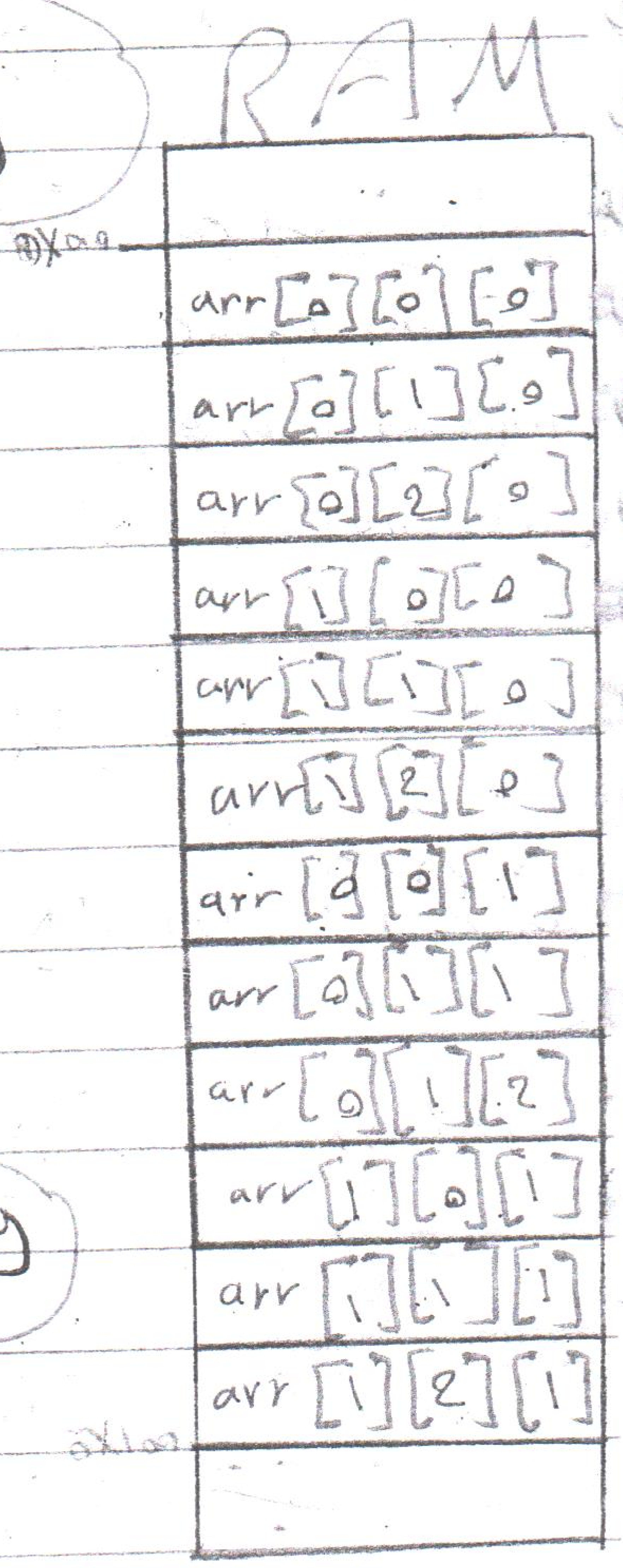
write C programe of 2D array and over acceiss it:

```
Hint int arr[2][3];
     arr[0][3] = 12;    printF(arr[1][0]);
```

## 3D Array

```
int arr[2][3][2]
   #Raws  #coloms  #
```




RAM

0x00

arr[0][0][0]
arr[0][1][0]
arr[0][2][0]
arr[1][0][0]
arr[1][1][0]
arr[1][2][0]
arr[0][0][1]
arr[0][1][1]
arr[0][1][2]
arr[1][0][1]
arr[1][1][1]
arr[1][2][1]

0x100

Use in handling 7-Sigmet display.

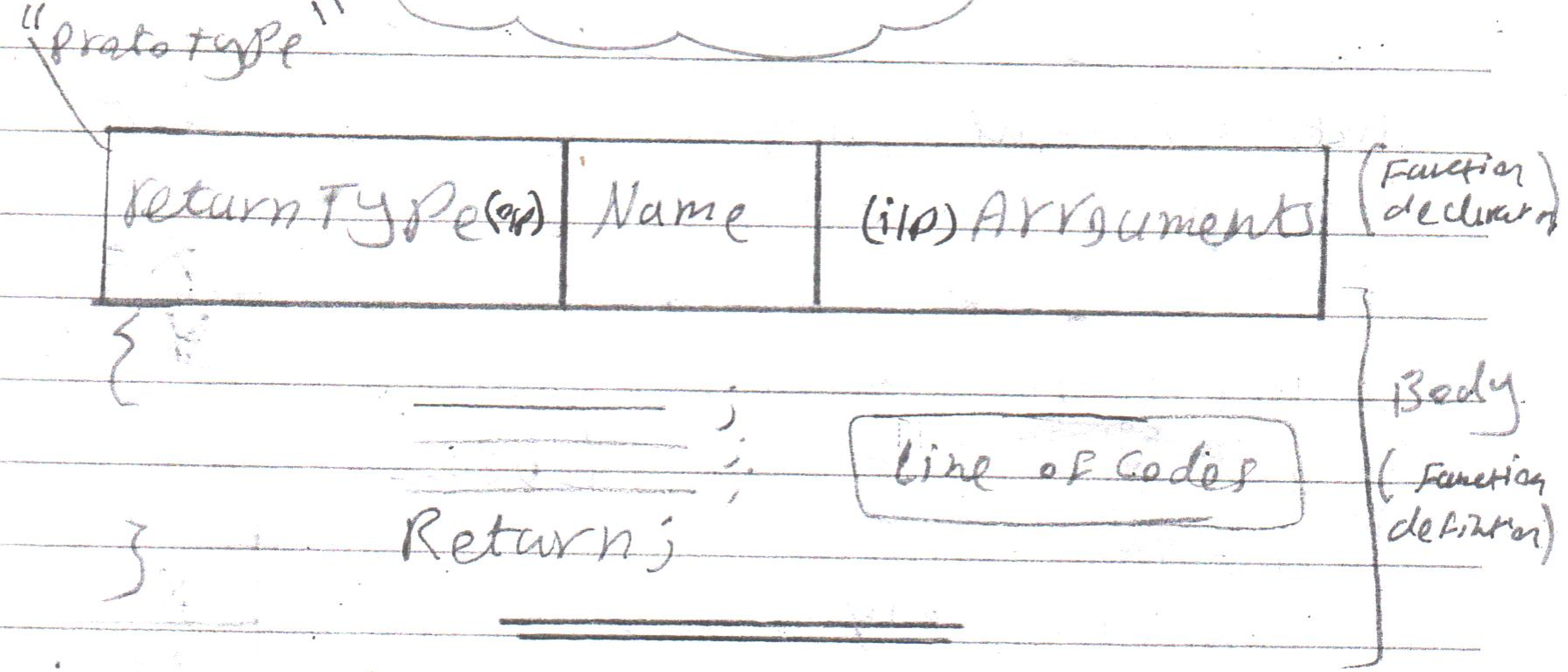
## Pointer as an array

```
int X;
int *ptr = &X;
    ptr[0] ⟶ = *ptr
    ptr[1]
    ptr[2]
```

/ The only defferent is array يحجز in memory but Ptr No.

PAGE
DATE 13/2/2016

# Functions



```
int  sum(int x ,int y)
{    int Resalt;
     Resalt= x + y;
      Return Resalt;
}
int main(. )
{    int Z;
   Z= sum(5,10);
   printf("%d",Z);
}
```

## Code 8 EX8:

write C code to Calcalate The Area of Cyrcle with Function.

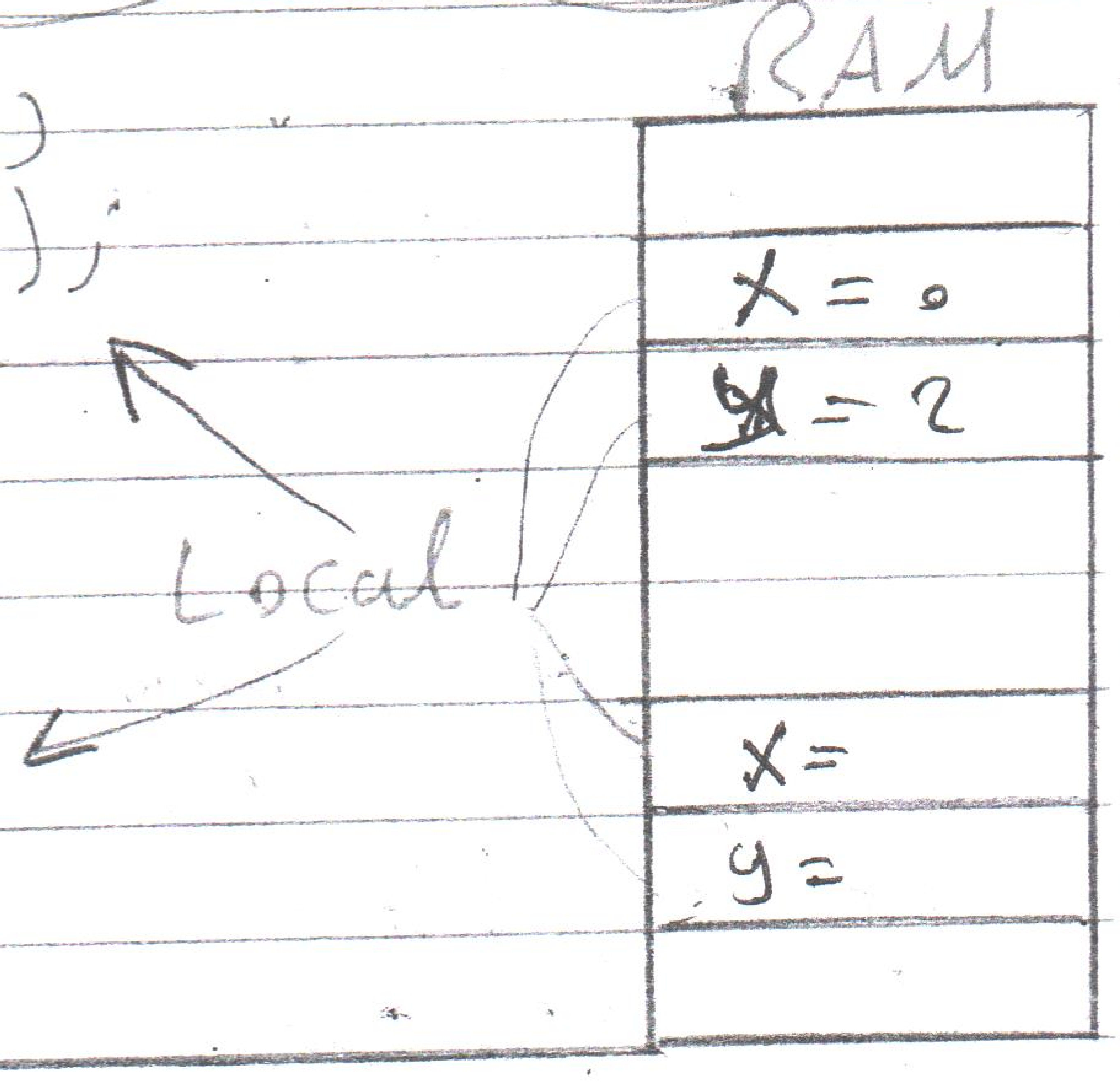
N.B

* ولو كتبت الـ Body كله بخير وبركة ، كتبت الـ Prototype بس
لازم تكون كامل الـ Definition في حته تانية في نفس الـ file

* ينفع تعمل multi declration بس Definition واحد بس

# Variable scope and life time

```
int sum( int x, int,y)
{      printf("%d", x);
}
int main( )
{   int x, y ;
      x = sum(5,10);
      y = sum(2,3);
}
```



[9] Code Exg:

write C.code of 2 function one to sum and the second to sub. all with x, y

## Global variables

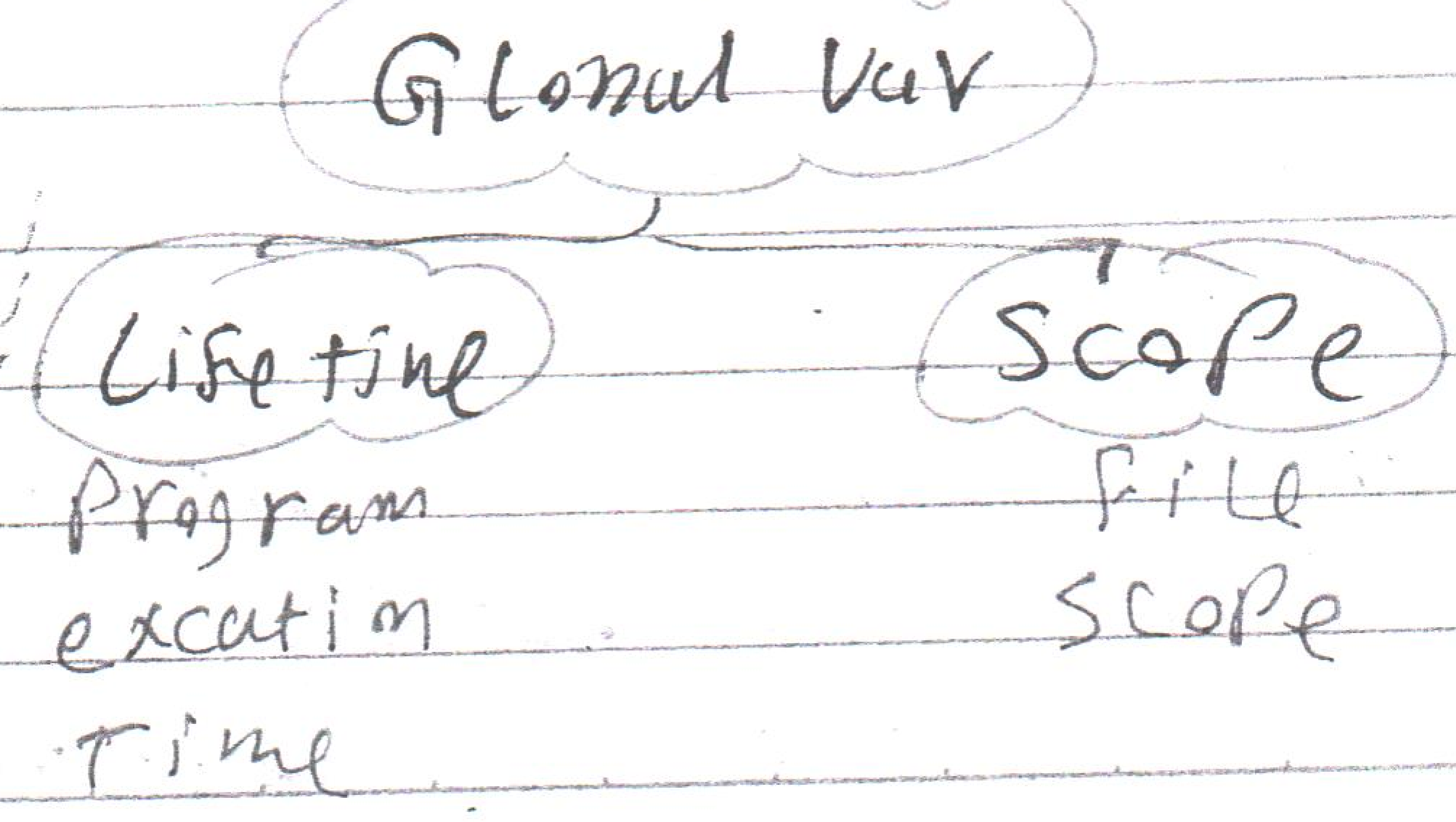
file scope, function scope
(inner) Block scope.

```
int x;

main( )
{


}
```

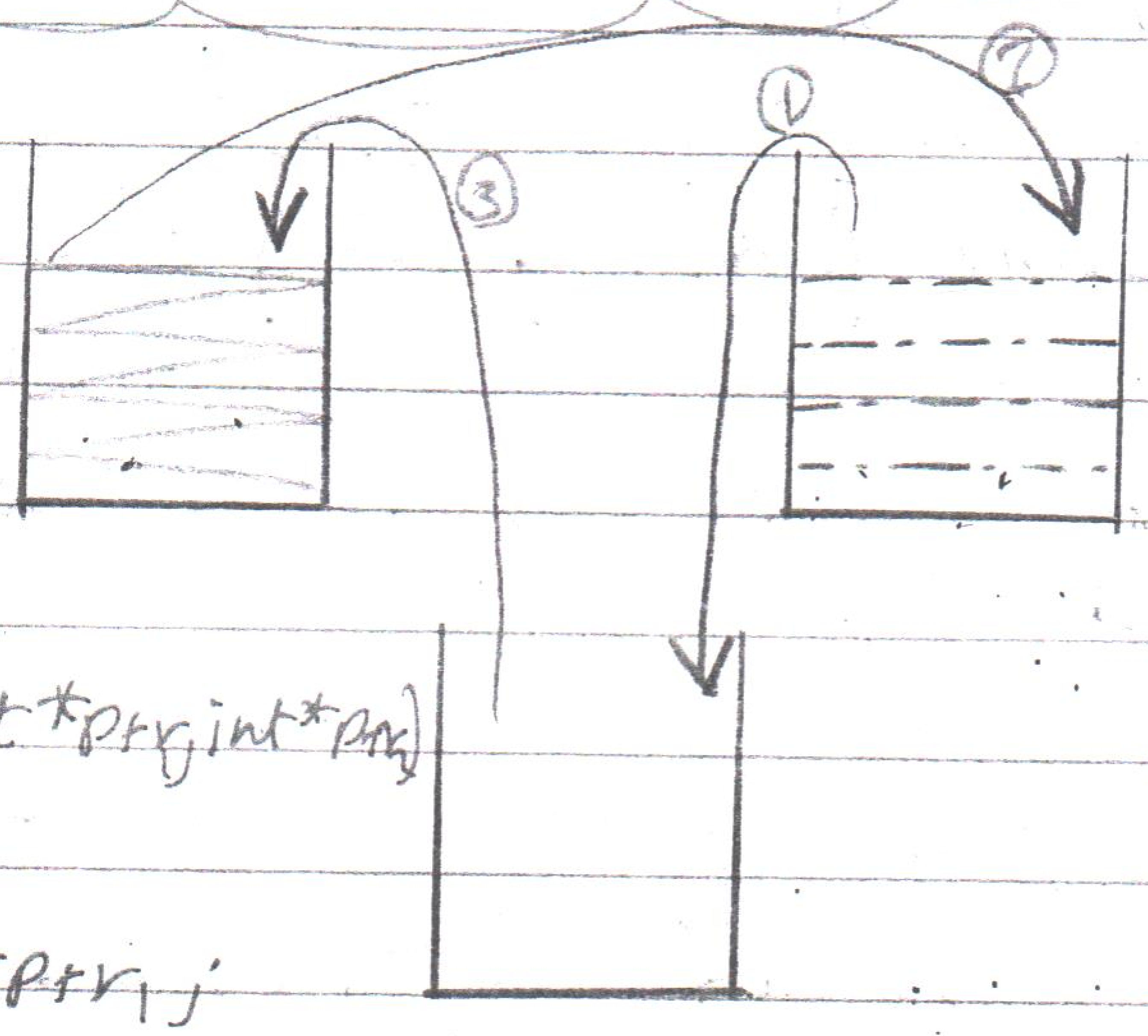
وظيفة ال Global Var زي ساعي
البريد بيحط ال Data عن الصندوقه
وبنت تروح تجيبها منه



* لو في Global var و local بنفس الاسم مش
هيدي error و بيشوف الأقرب ليه ( Local )

* لو عملت كذا Global بنفس الاسم        int x, x, x, x;
مش هيدي error بس تعمل initilization لواحدة بس

## Swap Function



```
void swap(int*ptr1,int*ptr2)
{ int Temp
  Temp=*ptr1;
  *ptr1=*ptr2;
  *ptr2=Temp;
}
```

## Headers

#include"

swap — header → prototype
swap — c → definition.

#incloude "swap.h"

19/2/2016

# Array to Function

```
void  func(int*ptr, int size)
 {
      ______
      ______
 }

void main(  )
{    int arr[5];
       Func(arr, 5);
}
```

* لل Array مبيتبعت غير By address

$$\frac{\text{sizeof(arr)}}{\text{sizeof(int)}}$$

## 10 Code Ex10:

write C code sends an array to func by address and print all of it.

# Structure

مكتبة

```
Struct Student
  {  unsigned char class;
     int Id;
     char grade;
  };
```

كدا لسه محجزتش حاجة في ال memorey

```
int main ( )
 { struct student Mohamed , Ali;

 }
```

كدة حجزت

PAGE

DATE

Cont. ① اول طريقة لل ~~definition~~ assignValue

```
Mohamed.Class=1;
Mohamed.Id     =1033;
Mohamed.grade='A';
Ali.Class      = 3;
Ali.Id         = 725;
Ali.grade      = 'B';
```

Dot operator ⇒ struct | → arrow oprator ⇒ ptr to struct

---

② تاني طريقة لل definition

```
struct student Mohamed = {1,1033,'A'};
```

③ تالت طريقة الـ definition

```
struct student Mohamed = {.Id=1033};
```

---

④ رابع طريقة لل defined

```
struct student
  {
      int seet;
      char name;
      float grad;
  } Mohamed={2,'A',3.2}, Ali={4,'B',4.7};
```

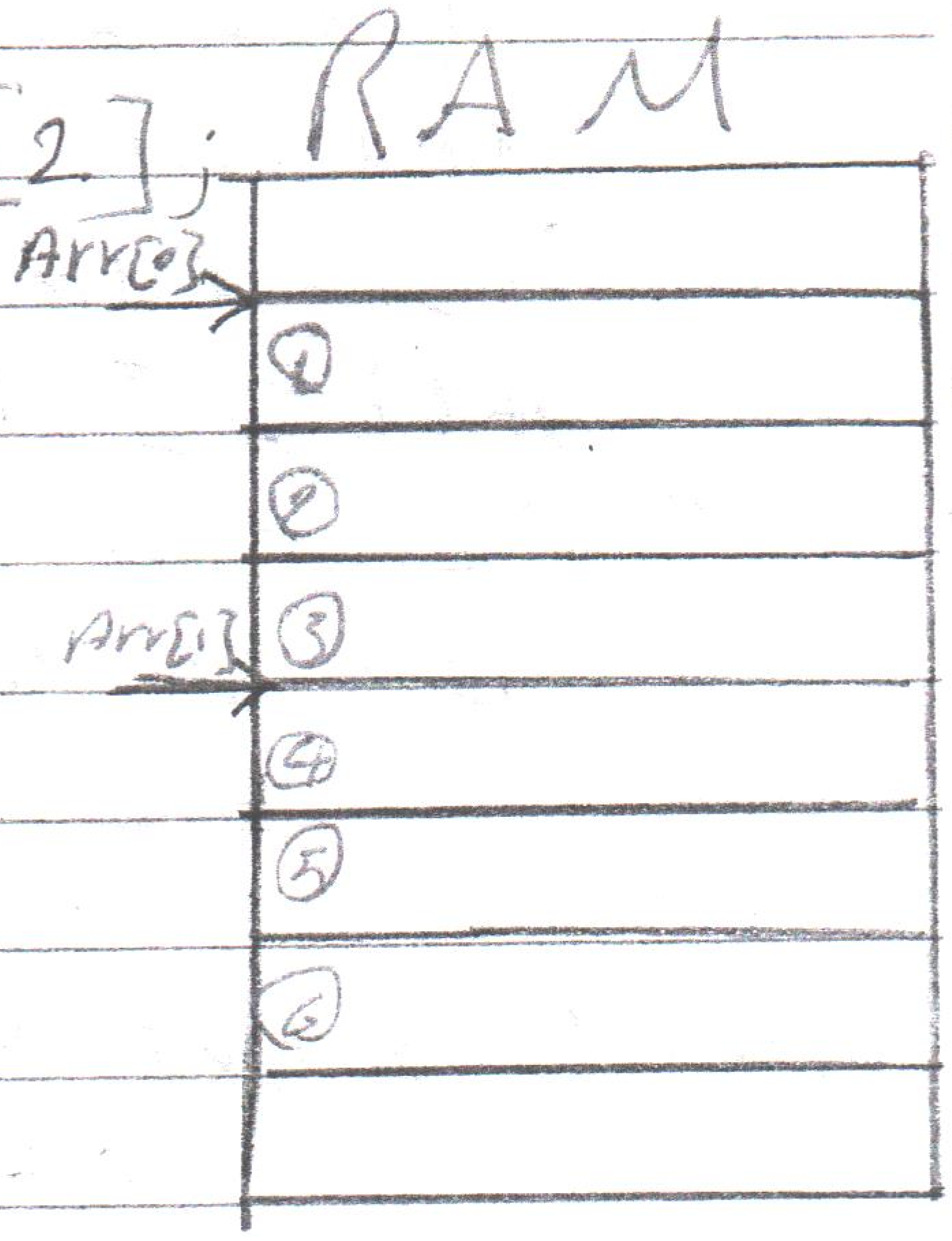
---

* الـ Body بتاع الـ Struct عبارة عن
الـ شبلونة أو الاسطمبة Stamp مش بتحجز حاجة
في الـ memory . مش بتحجز غير لما تـ Create منها

# Array of Struct;

```
main( )
{     struct student  Arr[2];

      Arr[0]. Class = 2 ;
      Arr[0]. grad= 3.4;
      Arr[0]. Nam= 'A';
      Arr[1] . Class  =  6  ;
      Arr [1] . grad  = 7.3 ;
      Arr[1]  . Name = 'B';
}
```



---

[11] Code EX 11:
write C code of Array of struct scan it from user and print it.

---

# TYPe def

```
TyPedef struct
            { int seet;
              char Name;
            } Student;      Not optinal
```

X Creat ; كل غايتها إننا نشيل ال Stract لما نيجي *
ب Creat ← New TyPe

Pointer to arrayof Struct.
ماشي

أول ما تسمع
الاسم ده أول
حاجة تيجي على بالك
→

# Pointer to struct

```
struct student Ali;
struct student *ptr = &Ali;
    ptr -> class = 2;
    ptr -> grad = 4.3;
```

Arrow aperator

ptr.class X
ptr->class ✓
ptr[3].class ✓
ptr[3]->class X

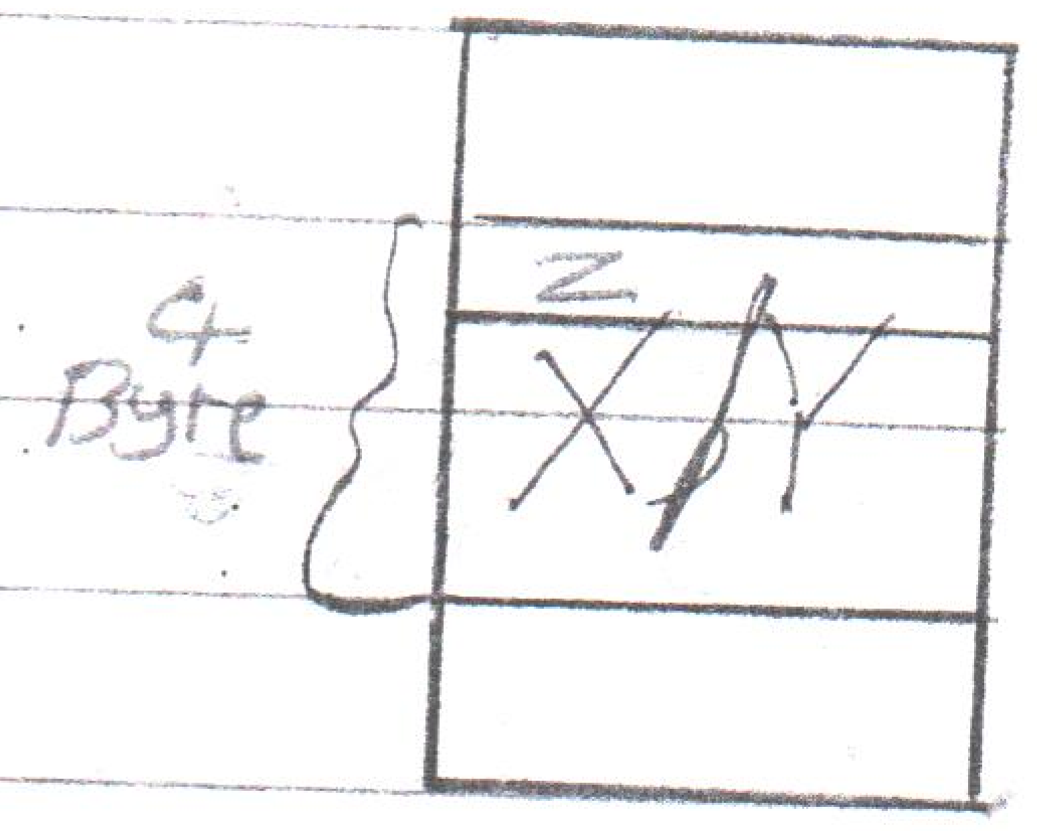
[12] Code EX12:

Write C Code of function That Takes struct by address and print it;

# Union

* exact Like struct. it reserve The biggest data type in it.

```
Union test
      {  int x;
         int y;
      }; char z;
```



20/2/2016

# Exponant & mantissa
الأس الكسر ([illegible])

float x = 10.375;

| Sign | exponant | mantissa |
|---|---|---|
| | | |

←1 Bit→ ← 8 Bit → ← 23 Byte →

1 30. 22 0

4 Byte
32 Bit

10.375

(Binery) 1010

| 0.375 | x2 | 0.75 | 0 |
|---|---|---|---|
| 0.75 | x2 | 1.5 | 1 |
| 0.5 | x2 | 1 | |
| 0 | | | |

1010.011

③ ② ①

Binery Base

1.010011 * 2

3 + 127 = 130 → exponant

| sign | exponent | mantissa |
|---|---|---|
| 0 | 130<br>10000010 | 010011 |

(N.13) if The divion is endless
∴ Float > double → in binary.

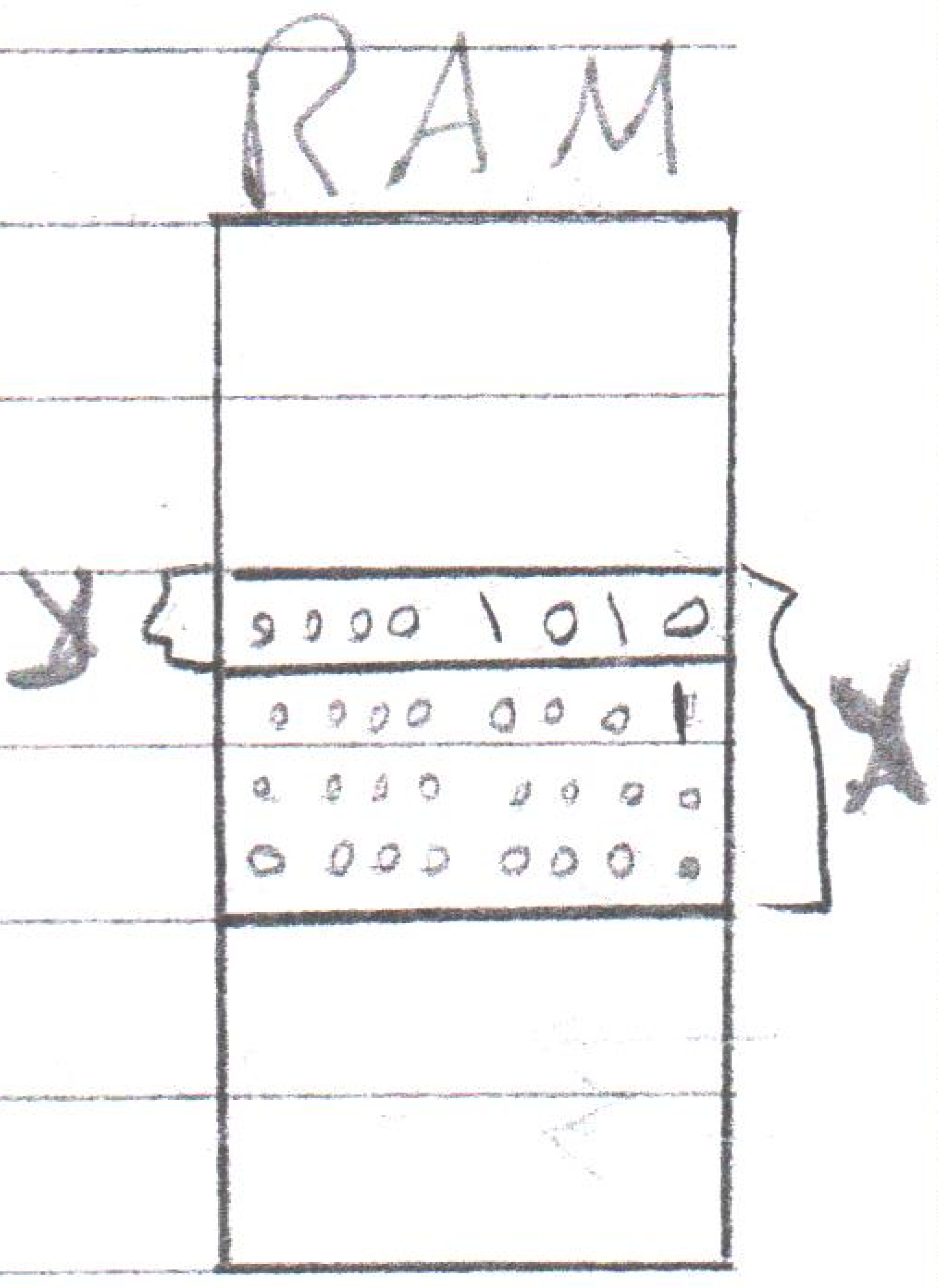
2.75 4 2 1 . 1/2 1/4 1/8
010.110

00

N.13

* اى حاجة متعرفة Global بـ inilaze بـ Zero .BSS
* اى حاجة متعرفة Local بـ inilaze بـ Rapfish
* ينفع اساوى struct بـ struct (بيساوى قيمهم)
* لما أقلك pointer to struct اول حاجة تيجى (فى) دماغك [→]
* ACCESS منها لو Pointer
* الفرقه بين [•] و [→]

# Union



```
Union test
      {   Char  Y;
          int   X;
      }
int main ( )
  {   union test My_Test;
      My_Test. X = 266;
      PrintF ("%d", My_Test. Y); → 10
  }
```

IT IS NOT CYCLEC Property

4 Byte( int )

|  |  | X | Y |
|---|---|---|---|
| 0000 0000 | 0000 0000 | 0000 0001 | 0000 1010 |

1 Byte(Char)

* لو عكست وحطيت قيمة Y 257 واطبعت الـ X هتعمل Cyclel Prop

# enam

```
                 0    1    2    3  =12  4 =13 5 =14 6 =15
enam week { Sat, san, mon, Tus, wed, Thr, fri };
int main(  )
 {    enam week Today ;
      Today = sat ;
      printf("%d", today);

 }
```

* لو عرفتله لـ element مثلاً "Sat=17" هيكمل العد بتسلسل
san=18, Mon=19 , Thar=20 ...

---

## مهم جداً جداً   Boolean   STring

* مفيش Boolean في الـ C-language بس ممكن نعمله بـ enum

```
→ enam Boolean { fals, True };
```

---

* مفيش STring في الـ C-language بس ممكن اعملها بالـ arr of char

```
→ char arr[ ] = "Mohamed";
→     printf(" %S ", arr);
```

* او ممكن نعملها هي هي بس من غير ما نحجز مكان في الـ memory:

```
→    char *ptr = "Mohamed";
→     printf("%S", ptr);
```

Sorting

# DATA STRUCTURE

## 1 Bubble_Sort.

2 For Loops "Nested"

Arr[10] = {12, 11, 13, 14, 2, 4, 0, 19, 15, 10}

```
    iF (Arr[i] > Arr[i+1])
        swap(Arr[i], Arr[i+1]);
```

```
              0    1    2    3    4    5    6    7    8    9
1-(0,1)  {11, 12, 13, 14, 2, 4, 0, 19, 15, 10}
2-(1,2)  {11, 12, 13, 14, 2, 4, 0, 19, 15, 10}
3-(2,3)  {11, 12, 13, 14, 2, 4, 0, 19, 15, 10}
4-(3,4)  {11, 12, 13, 14, 2, 4, 0, 19, 15, 10}
5-(4,5)  {11, 12, 13, 14, 2, 4, 0, 19, 15, 10}
6-(5,6)  {11, 12, 13, 14, 2, 0, 4, 19, 15, 10}
7-(6,7)  {11, 12, 13, 14, 2, 0, 4, 19, 15, 10}
8-(7,8)  {11, 12, 13, 14, 2, 0, 4, 15, 19, 10}
9-(8,9)  {11, 12, 13, 14, 2, 0, 4, 15, 10, 19}
```

---

* بعد ما لفوا لغاية الـ (size - 1) بقى اكبر رقم ف الآخر.

* محتاجين نكرر العملية دى بعدد الـ ( size ) بتاع الـ Arr.

* يعنى يبقى عدد الـ size-1 * size = 90 لفة فى المثال اللى فوق.

PAGE

DATE

```
#include<stdio.h>

 int Z;
Void swap(int*ptr1, int*ptr2)
{     int Temp;

      Temp = *ptr1;
      *ptr1 = *ptr2;
      *ptr2 = Temp;
}
Void bubble_sort(int*ptr, int size)
{
     int i,c,F;
     For(c=0;c<size;c++)
      {
            F=0;
            For(i=0;i<size-1;i++)
            {
                  Z++;
                 iF(ptr[i]>ptr[i+1])
                 {
                       swap(&ptr[i],&ptr[i+1]);
                 }
               else
                 {
                   F++;
                 }
            }
```

```
    iF(F ==size)
    {
        break ;
    }
}

For(i=0;i<size, i ++)
 {
        PrintF("%d\n", Ptr[i] ) ;
 }

 PrintF("\n\n\n %d ", z) ; //num of itraion
}

int main( )
{
  int arr[5] = {3,6,1, 7, 2};
  bubble_sort (arr, 5 );

}
```

## 2] Selection_ Sort

```
Void swap (int *ptr1, int *ptr2)
{
        int temp ;
        Temp = *ptr1 ;
        *ptr1 = *ptr2 ;
        *ptr2 = temp ;
}
int get_min_index (int *ptr, int start, int end)
{
    int   i  ;
    int min_index = start;
    For(i = start; i < end ; i ++)
     {      if (ptr [i] < ptr [min_index ])
                min_index = i  ;
     }
    return  min_index ;
}
Void Selection_sort (int *ptr, int size)
 {  int    i   ;
    int  min_index_value   ;
    For ( i = 0; i < size ; i ++)
 { min_index_value = get_min_index (ptr, i, size);
   swap (& ptr[i], & ptr[min_index_value]);
  }
}
```

```
int main ( )
{   int i j
    int arr [10] = {5,33,16,2,54,0,3,7,8};
 Selection_sort (arr, 10);
   For (i=0; i<10; i++)
     {
        Printf ("%d\n", arr[i]);
     }
}
```

## Searching

### [3] Linear Search

```
arr[10] = {1, 3, 0, 7, 12, 2, 5, 4, 6, 8}
For ( i = 0 ; i < size ; i ++)
 {
      if ( arr [i] == value )
      {   return i ;
      }
 }
```

PAGE

DATE

## [٤] Binary_Search

*لازم تكون مترتبة*

* what is The recursive fanction ?
itis fanction call itself.

```
int binary_search(int*Ptr, int Low, int hign, int Value)
{   int mid = (hign + Low) /2;
    if (Ptr [mid] == Value)
    {     return mid ;
    }
    else if (Ptr[mid] > Value )
    {  return binary_search (Ptr, Low, mid-1, Value)
    }
    else
    {   return binary serch (Ptr, mid+1, hign, Value)
    }
}

int main ( )
{   int X , result ;
    int arr [10] = {1, 3, 4, 5, 6, 7, 9, ١٠, ١١, ٢٠};
    printf ("enter number \n") ;
    scanf ( "%d", &X)
    result = binary_search (arr, 1, 10, X );
    printf ( "%d\n", result ) ;
}
```

ال return هترجع واحدة واحدة لحاية ماترجع لأول واحدة

## 5] Stack

```
int arr[10];
int c =0;
Void push(int x)
{      if (c!=9)
       {   arr[c]=x;
       }    c++;
       else
       {  printf("stack is full\n");
       }
}
int pop (void)
{    c--;
   if (c==-1)
   { printf("stack is empty\n");
   }  c=0;
  else
   {      return arr [c];
   }
int main ( )
{   int x;
    push(44);
   push(60);
   push(30);
 x=pop( );
                         printf("%d\n",x);
                         push(7);
                         push(9);
                       x=pop( );
                        printf("%d\n", x );
```

PAGE
DATE

## 6 queue

```
int arr[10];
int c =0 ;
void add (int x )
{   if(c!=10)
    {  arr[c]=x ;
    }  c ++;
    else
    {    printF("queue is fall\n") ;
    }
}
int get(void )
{       int temp =arr[0] , i ;
      For(i=0;i<c;i++)
      {     arr[i]=arr [i+1];
      }
      c--;        return temp;
}
int main( )
{    int x ;
   add (5);
   add (4);
   add (9);
   add (12);
}  x=get( ) ;
```

```
printF("%d\n",x)
x=get( );
printf("%d\n",x)
x=get( );
printF("%d\n",x)
```

# 7 Linked List

```
struct node
{    int  key;
     int  data;
     struct node *PNext;
};
struct node *pstart;
struct node *plast;
struct node New_node(void)
{  struct  node *pNew;
     PNew = ( struct node*)malloc(sizeof(struct node));
 printF("enter key\n");
 scanF ("%d", &PNew->key);
printF(" enter data\n");
scanF (" %d", &PNew->data);
PNew -> PNext = Null
return  pNew;
}
Void  add_Last (Void)
{    struct node *pNode = New_Node ( );
     if (pstart ==Null )
     {     pstart = Plast = pNode;
     }
```

```
   else
  {  plast->pNext = pNode;
     plast = pNode ;
     plast->p.Next=Null;
  }
}
void display_all (void)
{   struct node *pdisplay = pstart;
   while (pdisplay != Null)
        { printf(" %d \n", pdisplay->key);
          printf(" %d \n", pdisplay->data);
          pdisplay=>pdisplay=>pNext;
        }
}
struct node *Search(int value)
{      Struct node* pSearch = pStart;
     While (pSearch != Null)
            if (psearch->key == value)
            {  return pSearch;
            }
            else
            {     pSearch = pSearch->PNext;
            }
}
```

```
Void display_node (int value )
      Struct  node* ptr;
      ptr = search (Value);
     if  (ptr==Null)
     {   printf("net  found \n");
     }
    else
    {    printf(" %d \n", ptr->data);
    }
}
Void delete (Void)
{      int Temp;
     Temp = pstart->pNext;
      free  (pstart);
      pstart = temp;
}
 int  main(  )
{   int X, y;
    while  (1);
    {       printf("1=add new\n 2=printall\n 3=Search4.
                                                   delete"
         scanf("%d", & X);
         fflush (Stdin);
         switch (X)
        {   Case 1:
                     add_last( );
                 break ;
```

```
  case 2:
            display_all();
        break;
 case 3:
        printf("enter key to search\n");
        scanf("%d",&y);
        display_node(y);
        fflush(stdin )
      break;
case 4:
          delete();
     break;
default:
          printf("onley enter 1,2,3 or4");
          break;
          }
     }
}
```

# Dynamic memory allocation

* How to resirve in the life time

1- malloc( );
2- calloc( );
3- free ( );
4- Realloc( );

RESERVE IN HEAP

## 1 malloc

```
void* malloc(# of Bytes);
```

```
Ptr = (struct*) malloc( size of(struct) );
```

* explecit casting.

* Func بتاخد عدد ال Bytes و بترجع Void Pointer بيشاور على أول اللى أنا حجزته .

## 2 calloc

```
void* calloc(#of element, # of Bytes);
```

* الفرق بين ال malloc وال calloc

① ال malloc بتاخد #of Bytes بس ، لكن ال calloc بتاخد #of elements و #of Bytes

② ال malloc بتحط ف اللى بتحجزه RUppesh ، لكن ال calloc بتحط Zeros

## 3 free

```
free(Ptr);
```

بدلها ال ptr اللى أدهولى ال malloc أو calloc وهى بتشيل الحجز

## 4 realloc

```
realloc(Ptr, size الحجم الجديد);
```

بتعدل على الحجم القديم ف ال size